EXHIBIT LL

TO DECLARATION OF S. MERRILL WEISS IN SUPPORT OF PLAINTIFF ACACIA MEDIA TECHNOLOGIES CORPORATION'S MEMORANDUM OF POINTS AND AUTHORITIES IN OPPOSITION TO ROUND 3 DEFENDANTS' MOTION FOR SUMMARY JUDGMENT OF INVALIDITY UNDER 35 U.S.C. § 112 OF THE '992, '863, AND '702 PATENTS; AND SATELLITE DEFENDANTS' MOTION FOR SUMMARY JUDGMENT OF INVALIDITY OF THE '992, '863, AND '720 PATENTS

Hovsepian v. Apple, Inc. Doc. 325 Att. 40

Dockets.Justia.com

Chapter 1

An Introduction to Reed-Solomon Codes

Stephen B. Wicker
School of Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332

Vijay K. Bhargava
Department of Electrical and Computer Engineering
University of Victoria
Victoria, British Columbia, Canada V8W 2Y2

1 POLYNOMIAL CODES OVER CERTAIN FINITE FIELDS

On January 21, 1959, Irving Reed and Gus Solomon submitted a paper to the *Journal of the Society for Industrial and Applied Mathematics.* In June of 1960 the paper was published: five pages under the rather unpretentious title "Polynomial Codes over Certain Finite Fields" [17]. This paper described a new class of error-correcting codes that are now called Reed-Solomon codes. In the decades since their discovery, Reed-Solomon codes have enjoyed countless applications, from compact disc™ players in living rooms all over the planet to spacecraft that are now well beyond the orbit of Pluto. Reed-Solomon codes have been an integral part of the telecommunications revolution in the last half of the twentieth century. This book has been written in an attempt to capture the power and utility of these codes, as well as the history of their use. Each chapter has been written by specialists in digital communications who have used Reed-Solomon codes in their engineering designs or have made the analysis and implementation of Reed-Solomon codes a focus of their research. The chapters can be loosely grouped into four classes: history, code construction, applications, and decoding techniques. The chapter immediately following this introduction is a joint effort by Irving Reed and Gus Solomon. In this chapter they describe the events that led up to the discovery of Reed-Solomon codes and their experiences afterwards. It is basically historical in nature and provides some excellent insights into a great event in the history of digital communications technology. The remaining chapters deal with Reed-Solomon codes in a more detailed, technical manner. It is thus appropriate that the reader be prepared with a bit of introductory material. The remainder of this chapter presents the three basic techniques for constructing Reed-Solomon codes and briefly discusses some typical applications and the decoding problem. In the process, the remaining chapters in this book are introduced.

2 THE ORIGINAL APPROACH TO REED-SOLOMON CODES

Reed-Solomon codes are constructed and decoded through the use of finite field arithmetic. Finite fields were the discovery of the French mathematician Evariste Galois and are thus sometimes referred to as Galois fields. A finite field of q elements is usually denoted as GF(q). The number of elements in a finite field must be of the form p^m, where p is a prime integer and m is a positive integer [12, 21]. For any given q of this form, the field GF(q) is unique up to isomorphisms (in other words, one can rename the elements in the field, but it is still the same field). We can thus completely describe a finite field by giving its size.

The *order* of an element α in GF(q) is the smallest positive integer m such that $\alpha^m = 1$. GF(q) always contains at least one element, called a primitive element, that has order $(q-1)$. Let α be primitive in GF(q). Since $(q-1)$ consecutive powers of α, $\{1, \alpha, \alpha^2, \ldots, \alpha^{q-2}\}$, must be distinct, they are the $(q-1)$ nonzero elements of GF(q). The "exponential representation" of the nonzero elements in the field provides an obvious means for describing the multiplication operation: $\alpha^x \cdot \alpha^y = \alpha^{(x+y)}$.

Addition in GF(q) is almost as easy. A primitive element is a root of a *primitive polynomial* $p(x)$. The exponential representations for the nonzero elements of GF(q) are reduced *modulo* the primitive polynomial to obtain a "polynomial representation," which is used in the addition operation.

EXAMPLE: GF(8) $p(x) = x^3 + x + 1$ is a primitive binary polynomial. Let α be a root of $p(x)$. This implies that $\alpha^3 + \alpha + 1 = 0$, or equivalently, $\alpha^3 = \alpha + 1$ (addition and subtraction are the same in binary arithmetic).

Exponential Representation		Polynomial Representation
1	$=$	1
α^1	$=$	α
α^2	$=$	α^2
α^3	$=$	$\alpha + 1$
α^4	$=$	$\alpha^2 + \alpha$
α^5	$=$	$\alpha^3 + \alpha^2 = \alpha^2 + \alpha + 1$
α^6	$=$	$\alpha^3 + \alpha^2 + \alpha = \alpha^2 + 1$
0	$=$	0

Addition is performed using the polynomial representation. To compute $\alpha^2 + \alpha^5$ in GF(8), one begins by substituting the polynomial representations for the exponential representations α^2 and α^5. The polynomials are then summed to obtain a third polynomial representation, which may then be reexpressed as a power of α.

$$\alpha^2 + \alpha^5 = (\alpha^2) + (\alpha^2 + \alpha + 1) = (\alpha + 1) = \alpha^3$$

The original approach to constructing Reed-Solomon codes is extremely simple (brilliant ideas usually are). Suppose that we have a packet of k information symbols, $\{m_0, m_1, \ldots, m_{k-2}, m_{k-1}\}$, taken from the finite field GF(q). These symbols can be used to construct a polynomial $P(x) = m_0 + m_1 x + \cdots + m_{k-2}x^{k-2} + m_{k-1}x^{k-1}$. A Reed-Solomon code word $\mathbf{c}$ is formed by evaluating $P(x)$ at each of the q elements in the finite field GF(q).

$$\mathbf{c} = (c_0, c_1, c_2, \ldots, c_{q-1}) = \left[P(0), P(\alpha), P(\alpha^2), \ldots, P(\alpha^{q-1})\right] \quad (1)$$

A complete set of code words is constructed by allowing the k information symbols to take on all possible values. Since the information symbols are selected from GF(q), they can each take on q different values. There are thus q^k code words in this Reed-Solomon code. A code is said to be linear if the sum of any two code words is also a code word. It follows from Equation (1) that Reed-Solomon codes are linear, for the sum of two polynomials of degree $(k-1)$ is simply another polynomial of degree less than or equal to $(k-1)$.

The number of information symbols k is frequently called the *dimension* of the code. This term is derived from the fact that the Reed-Solomon code words form a vector space of dimension k over GF(q). Since each code word has q coordinates [see equation (1)], it is usually said that the code has *length* $n = q$. When Reed-Solomon codes (and any other linear codes) are discussed, they are usually denoted by their length n and dimension k as (n, k) codes.

Each Reed-Solomon code word can be related by equation (1) to a system of q linear equations in k variables, as shown below.

$$\begin{aligned}
P(0) &= m_0 \\
P(\alpha) &= m_0 + m_1\alpha + m_2\alpha^2 + \cdots + m_{k-1}\alpha^{k-1} \\
P\left(\alpha^2\right) &= m_0 + m_1\alpha^2 + m_2\alpha^4 + \cdots + m_{k-1}\alpha^{2(k-1)} \\
&\vdots \\
P\left(\alpha^{q-1}\right) &= m_0 + m_1\alpha^{q-1} + m_2\alpha^{2(q-1)} + \cdots + m_{k-1}\alpha^{(k-1)(q-1)}
\end{aligned} \quad (2)$$

Any k of these expressions can be used to construct a system of k equations in k variables. For example, the first k of the above expressions form the following system.

$$\begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 1 & \alpha & \alpha^2 & \cdots & \alpha^{(k-1)} \\ 1 & \alpha^2 & \alpha^4 & \cdots & \alpha^{2(k-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha^{k-1} & \alpha^{2(k-1)} & \cdots & \alpha^{(k-1)(k-1)} \end{bmatrix} \cdot \begin{bmatrix} m_0 \\ m_1 \\ m_2 \\ \vdots \\ m_{k-1} \end{bmatrix} = \begin{bmatrix} P(0) \\ P(\alpha) \\ P(\alpha^2) \\ \vdots \\ P(\alpha^{k-1}) \end{bmatrix} \quad (3)$$

This system can be shown to have a unique solution for the k information symbols $\{m_0, m_1, \ldots, m_{k-2}, m_{k-1}\}$ by computing the determinant of the following coefficient matrix.

$$\begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 1 & \alpha & \alpha^2 & \cdots & \alpha^{(k-1)} \\ 1 & \alpha^2 & \alpha^4 & \cdots & \alpha^{2(k-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha^{k-1} & \alpha^{2(k-1)} & \cdots & \alpha^{(k-1)(k-1)} \end{bmatrix} \quad (4)$$

Using cofactor expansion across the top row, the determinant of this matrix reduces to that of a *Vandermonde* matrix, and it can be shown that all Vandermonde matrices are nonsingular [21]. The coefficient matrix for any system formed by a combination of k expressions from equation (2) can be reduced to Vandermonde form, so it follows that any k of the expressions in equation (2) can be used to determine the values of the information coordinates.

We are now in a position to evaluate the ability of the Reed-Solomon code to correct errors. Suppose that t of the code word coordinates are corrupted by noise during transmission and received incorrectly. The corresponding expressions in equation (2) are thus incorrect and would lead to an incorrect solution if one or more of them were used in the system of equations in (3). Assuming that we do not know where the errors are, we might construct all possible distinct systems of k expressions from the set of expressions in equation (2). There are $\binom{q}{k}$ such systems, $\binom{t+k-1}{k}$ of which will give incorrect information symbols [17]. If we take the majority opinion among the solutions to all possible linear systems, we will get the correct information bits so long as $\binom{t+k-1}{k} < \binom{q-t}{k}$. This condition holds if and only if $t + k - 1 < q - t$, which in turn holds if and only if $2t < q - k + 1$. A Reed-Solomon code of length q and dimension k can thus correct up to t errors, where t is as follows. Note that $\lfloor x \rfloor$ is the largest integer less than or equal to x.

$$t = \left\lfloor \frac{q - k + 1}{2} \right\rfloor \quad (5)$$

In 1964 Singleton showed that this was the best possible error correction capability for any code of the same length and dimension [18]. Codes that achieve this "optimal" error correction capability are called *maximum distance separable* (MDS). Reed-Solomon codes are by far the dominant members, both in number and utility, of the class of MDS codes. MDS codes have a number of interesting properties that lead to many practical consequences. This is discussed in some detail in Chapter 7.

We have shown that erroneous expressions in equation (2) can be corrected up to some maximum number of errors. But what if some of the expressions in equation (2) are missing altogether? In many digital communication systems, the demodulator can make a rough estimate as to whether a given symbol at the output of the detector is reliable. For example, a binary detector might consist of a simple hard limiter: analog received signals below a certain threshold are made into zeros, those above the threshold become ones. If we know that a particular signal is very close to the threshold, we may want to declare that signal as being "erased" instead of assigning a binary value that has a significant probability of being incorrect. When we erase a Reed-Solomon code word coordinate, we are deleting the corresponding expression in equation (2) from consideration. Since we need only k correct expressions to recover the information bits, we can erase up to $q - k$ of the code word coordinates. Combining this result with the above result for error correction, it can be shown that a Reed-Solomon code can correct t errors and v erasures so long as

$$2t + v < q - k + 1. \tag{6}$$

Reed and Solomon's original approach to constructing their codes fell out of favor following the discovery of the "generator polynomial approach" (the subject of the next section). At the time it was felt that the latter approach led to better decoding algorithms. It was not until 1982 that Tsfasman, Vladut, and Zink, using a technique developed by Goppa, extended Reed and Solomon's construction to develop a class of codes whose performance exceeded the Gilbert-Varshamov bound [20]. The Gilbert-Varshamov bound is a lower bound on the performance of error-correcting codes that many were beginning to believe was also an upper bound. Tsfasman, Vladut, and Zink's work uses several powerful tools from algebraic geometry to break open an entirely new field of research that continues to attract a great deal of interest from coding theorists. Yaghoobian and Blake explore this field in detail in Chapter 13. They begin by considering Reed and Solomon's construction from an algebra-geometric perspective. The Galois field elements $\{0, \alpha, \alpha^2, \ldots, \alpha^{q-1} = 1\}$ are treated as points on a rational curve; along with the point at infinity, they form the one-dimensional *projective line*. The operation in equation (1) translates these points onto points on a curve in a higher-dimensional projective space. Yaghoobian and Blake show that by changing the "base" curve from the projective line to a curve with more structure (i.e., one with higher genus), codes defined by the transformation in equation (1) can yield new codes that are extremely powerful.

3 THE GENERATOR POLYNOMIAL APPROACH

The generator polynomial construction for Reed-Solomon codes is the approach most commonly used today in the error control literature. This approach initially evolved independently from Reed-Solomon codes as a means for describing *cyclic codes.* A code is said to be cyclic if, for any code word $\mathbf{c} = (c_0, c_1, c_2, \ldots, c_{n-2}, c_{n-1})$, the cyclically shifted word $\mathbf{c}' = (c_1, c_2, c_3, \ldots, c_{n-1}, c_0)$ is also a code word. Cyclic codes were first discussed in a series of technical notes and reports written between 1957 and 1959 by Prange at the Air Force Cambridge Research Labs [14–16]. This led directly to the work published in March and September of 1960 by Bose and Ray-Chaudhuri on what are now called BCH codes [4, 5]. (The "H" in BCH is for Hocquenghem, whose 1959 paper presented independent work that included a description of BCH codes as a "generalization of Hamming's work" [9].) Gorenstein and Zierler then generalized Bose and Ray-Chaudhuri's work to arbitrary Galois fields of size p^m, discovering along the way that they had developed a new means for describing Reed and Solomon's "polynomial codes" [8].

If an (n, k) code is cyclic, it can be shown that the code can always be defined using a generator polynomial $g(x) = g_0 + g_1 x + g_2 x^2 + \cdots + g_{n-k} x^{n-k}$. In this definition each code word is interpreted as a *code polynomial.*

$$(c_0, c_1, c_2, \cdots, c_{n-1}) \Rightarrow c_0 + c_1 x + c_2 x^2 + \cdots + c_{n-1} x^{n-1} \tag{7}$$

A vector $\mathbf{c}$ is a code word in the code defined by $g(x)$ if and only if its corresponding code polynomial $c(x)$ is a multiple of $g(x)$. This provides a very convenient means for mapping information symbols onto code words. Let $\mathbf{m} = (m_0, m_1, \ldots, m_{k-1})$ be a block of k information symbols. These symbols can be associated with an information polynomial $m(x) = m_0 + m_1 x + \cdots + m_{k-1} x^{k-1}$, which is encoded through multiplication by $g(x)$.

$$c(x) = m(x) g(x) \tag{8}$$

Cyclic Reed-Solomon codes with code word symbols from GF(q) have length $q - 1$, one coordinate less than that obtained through the original construction. A cyclic Reed-Solomon code can be extended to have length q, or even length $q + 1$, but in both cases the resulting code is usually no longer cyclic [21]. As the generator polynomial approach to constructing

Reed-Solomon codes is currently the most popular, the reader may note that Reed-Solomon codes with symbols in the field GF(q) usually have length $q - 1$. For example, the field GF(256) is used in many applications because each of the 256 field elements can be represented as an 8-bit sequence, or byte. Reed-Solomon codes of length 255 are thus very popular error control codes (see the deep space coding standard in Chapters 3 and 11).

The cyclic Reed-Solomon code construction process proceeds as follows. Suppose that we want to build a t-error-correcting Reed-Solomon code of length $q - 1$ with symbols in GF(q). Recall that the nonzero elements in the Galois field GF(q) can be represented as $(q - 1)$ powers of some primitive element α. The Reed-Solomon design criterion is as follows: *The generator polynomial for a t-error-correcting code must have as roots 2t consecutive powers of* α.

$$g(x) = \prod_{j=1}^{2t} \left(x - \alpha^j\right) \tag{9}$$

Valid code polynomials can thus have degrees from $2t$ up to $q - 2$ [a degree-$(q - 2)$ code polynomial corresponds to a code word with $(q - 1)$ coordinates]. It follows that the dimension of a code with a degree-$2t$ generator polynomial is $k = q - 2t - 1$. Once again we see the MDS relation

$$\text{Error correction capability} = \frac{\text{length} - \text{dimension}}{2}. \tag{10}$$

Any valid code polynomial must be a multiple of the generator polynomial. It follows that any valid code polynomial must have as roots the same $2t$ consecutive powers of α that form the roots of $g(x)$. This provides us with a very convenient means for determining whether a received word is a valid code word. We simply make sure that the corresponding polynomial has the necessary roots. This approach leads to a powerful and efficient set of decoding algorithms that are introduced later in this chapter and discussed in detail in Chapters 5 and 10.

4 THE GALOIS FIELD FOURIER TRANSFORM APPROACH

The third approach to Reed-Solomon codes uses the various techniques of Fourier transforms to achieve some interesting interpretations of the encoding and decoding process. Once again, let α be a primitive element in the Galois field GF(q). The Galois field Fourier transform (GFFT) of an n-bit vector $\mathbf{c} = (c_0, c_1, \ldots, c_{n-1})$ is defined as follows.

$$\mathcal{F}\{(c_0, c_1, \ldots, c_{n-1})\} = (C_0, C_1, \ldots, C_{n-1}), \quad \text{where } C_j = \sum_{i=0}^{n-1} c_i\alpha^{ij},\ j = 0, 1, \ldots, n-1 \tag{11}$$

This is clearly a revisitation of Reed and Solomon's original approach, but using the vocabulary and power of Fourier transforms.

Unlike the conventional analysis of signals in a communication system, it is not entirely clear what is meant by the terms "time domain" and "frequency domain" when we are working with coordinate values from finite fields. Despite this bit of fogginess, we can press on to some very interesting and useful results. Suppose that our n-bit word in the time domain is a code word **c** from a cyclic, t-error-correcting Reed-Solomon code. **c** thus corresponds to a code polynomial that has as roots some $2t$ consecutive powers of α. When we take the GFFT of this n-bit word, we find that the frequency domain word, or *spectrum*, has $2t$ consecutive zero coordinates! It can be shown that the two conditions are equivalent: a word polynomial has $2t$ consecutive powers of α as roots if and only if the spectrum of the corresponding word has $2t$ consecutive zero coordinates. The GFFT approach is thus a dual to the generator polynomial approach. The transform relationship leads to a series of efficient encoders and decoders. The interested reader is referred to the pioneering work in [3].

5 APPLICATIONS OF REED-SOLOMON CODES

5.1 The Digital Audio Disc

It can safely be claimed that Reed-Solomon codes are the most frequently used digital error control codes in the world. This claim rests firmly on the fact that the digital audio disc, or compact disc uses Reed-Solomon codes for error correction and error concealment. In Chapter 4, Immink describes how digital audio systems make use of Reed-Solomon codes and how the special properties of Reed-Solomon codes make the sound quality of the compact disc as impressive as it is (the signal-to-noise ratio at the output exceeds 90 dB).

The compact disc system uses a pair of *cross-interleaved* Reed-Solomon codes. The details are left for presentation in Chapter 4, but three items of particular interest will be noted here. Since Reed-Solomon codes are nonbinary, each code word symbol becomes a string of bits when transmitted across a binary channel. If a noise burst corrupts several consecutive bits on the channel, the resulting bit errors are "trapped" within a small number of nonbinary symbols. For each burst of noise, the Reed-Solomon decoder needs only to correct a few symbol errors, as opposed to a longer string of bit errors.

Reed-Solomon codes can also correct erasures in an efficient manner. The compact disc error control system uses the first of the cross-interleaved

Reed-Solomon codes to declare erasures, and the second code to correct them. This brilliant piece of engineering allows for accurate reproduction of sound despite material imperfections and damage to the surface of the disc.

Finally, Reed-Solomon codes, as with any block code used in a systematic context, can be "shortened" to an arbitrary extent. Most error control codes have a natural length. For example, Reed-Solomon codes defined over the field GF(256) can be said to have a natural length of 256 (the original approach) or 255 (the generator polynomial approach). The compact disc system is able to use Reed-Solomon codes of length 32 and 28 symbols, while still retaining the 8-bit symbol structure of a length-255 code. The shorter code word length keeps the implementation of the interleaver simple, while the 8-bit symbols provide protection against error bursts and provide a good match for the 16-bit samples taken from the analog music source.

5.2 Deep Space Telecommunication Systems

It has been said that deep space telecommunications and coding are a "match made in heaven" [11]. For Reed-Solomon codes this has certainly been the case. In Chapter 3 McEliece and Swanson examine the use of Reed-Solomon codes in several of NASA and ESA's planetary exploration missions. They begin by noting that Reed-Solomon codes were not an obvious choice for deep space telecommunication systems because the deep space channel does not usually induce burst errors in transmitted data. It was soon found, however, that when convolutional and Reed-Solomon codes are used in concatenated systems, enormous coding gains are achievable. A convolutional code is used as an "inner code," while a Reed-Solomon code is used to correct errors at the output of the convolutional (Viterbi) decoder. The Viterbi decoder output happens to be bursty, providing a perfect match for a Reed-Solomon code. The most famous application of the concatenated convolutional/Reed-Solomon system was in the *Voyager* expeditions to Uranus and to Neptune. Reed-Solomon codes were used in the transmission of photographs from these outer planets, providing close-up images of worlds that, to the residents of this planet, were once tiny smudges made visible only through the use of powerful telescopes. Chapter 3 contains the first photograph of Uranus taken by *Voyager*, a photograph that is also the first Reed-Solomon-encoded image ever transmitted from deep space.

Chapter 3 also discusses the problems encountered by the *Galileo* mission to Jupiter. The high-gain antenna on board the spacecraft has refused to deploy properly and is thus useless. All data collected by the probe must now be sent to the earth by way of a low-gain antenna, resulting in a drastic lowering of the rate at which the spacecraft can reliably transmit data. Engineers all over the world have been frantically working to find ways to increase the coding gain provided by the concatenated codes used by *Galileo*. In Chapter 11, Hagenauer, Offer, and Papke discuss several powerful means for attacking this problem. These include the use of iterative decoding and the soft-output Viterbi algorithm (SOVA). The SOVA provides for the declaration of erasures at the input to the Reed-Solomon decoder, thus improving performance considerably. One key result that emerges from this work is the demonstration that a concatenated convolutional/Reed-Solomon system can provide for reliable data transmission beyond the *cutoff rate*. The cutoff rate is an information-theoretic concept that many believe denotes the best possible performance for an error control system. The results in Chapter 11 show that, for a concatenated error control system, this supposed barrier can be surpassed. Hagenauer, Offer, and Papke have brought the performance of a Reed-Solomon error control system within a few decibels of the ultimate barrier: channel capacity. A series of photos of the lunar surface are used to display the impact of these results.

5.3 Error Control for Systems with Feedback

In Chapter 7, Wicker and Bartz examine various means for using Reed-Solomon codes in applications that allow the transmission of information from the receiver back to the transmitter. Such applications include mobile data transmission systems and high-reliability military communication systems. Along with their powerful error correction capabilities, Reed-Solomon codes can also provide a substantial amount of *simultaneous* error detection. The key lies in the distinction between a *decoder error* and a *decoder failure*. Consider the decoding of a t-error-correcting Reed-Solomon code. If a received word differs from an incorrect code word in t or fewer coordinates, then that code word will be selected by the decoder, resulting in a decoder error. This is an undetectable condition that causes errors in the decoded data. On the other hand, if a noise-corrupted word differs from *all* code words in $(t + 1)$ or more coordinates, then the decoder declares a *decoder failure*. Decoder failures are detectable, so the receiver is able to request a retransmission of the problematic word. In Chapter 7 it is shown that decoder errors are actually quite rare and that retransmission requests can be used to develop Reed-Solomon error control systems with extremely high levels of reliability. Two such systems are discussed. The first is a simple extension of the standard forward-error-correcting Reed-Solomon error control system. The second system uses the special properties of Reed-Solomon codes to a create a *code-combining system*. Multiple received words are combined to create code words from increasingly powerful Reed-Solomon codes. No matter how noisy the channel may be, this system ensures that reliable data will eventually be provided to the end user.

5.4 Spread-Spectrum Systems

Spread-spectrum systems can be grouped into two basic types: frequency-hopping spread spectrum (FH/SS) and direct-sequence spread spectrum (DS/SS). An FH/SS system modulates information onto a carrier that is systematically moved from frequency to frequency. Frequency hopping has been used in military communications systems as a powerful means for defeating partial-band jamming. In general a frequency-hopped system provides protection against any partial-band disturbance. In a peaceful environment, such disturbances may arise from FH multiple-access interference or from narrowband noise sources. They may also be caused by frequency selective fading on a mobile communication channel. For these reasons and others, frequency hopping is receiving serious consideration for use in personal communication systems in the United States. GSM, the European cellular telephone system, has already provided for a limited amount of frequency hopping.

A DS/SS system creates a wideband signal by phase-shift-keying its RF carrier with the sum of the data sequence and a spreading sequence whose pulse rate is much larger than that of the data sequence. When the received signal is "despread," narrowband interfering signals are spread out so that only a small fraction of their power falls within the bandwidth of the recovered data sequence. As with FH/SS systems, DS/SS systems perform well against partial-band disturbances. DS/SS systems have the added benefit of a low-probability-of-intercept (LPI) spectrum. Since the transmitted signal energy is spread across a wide spectrum, only a small amount of signal energy is found in any given narrowband slot in which an enemy may be searching for activity. DS/SS systems are also being considered for use in mobile radio applications.

In Chapter 9, Sarwate begins by describing the design and performance of several FH/SS systems. He then proceeds to discuss how Reed-Solomon codes can be used in the design of the hopping sequences. If these sequences are carefully selected, the interference caused by other users in a multiple-access environment can be greatly reduced. Sarwate shows that some of the hopping sequences that have been described using other terminology can also be viewed as low-rate Reed-Solomon code words. He then proceeds to a description of several DS/SS systems and shows that the familiar Gold and Kasami sequences can also be interpreted using the language of Reed-Solomon codes.

In Chapter 8, Pursley explores in greater detail the application of Reed-Solomon codes for error correction in FH/SS systems. He focuses on various means by which side information may be obtained from a frequency-hopped system and used to declare erasures at the input to the Reed-Solomon decoder. He begins by considering the use of test symbols and threshold tests, which require very little modification to an existing FH/SS system but provide a significant improvement in performance. He then considers more powerful techniques, including the use of binary parity checks on individual code word symbols and the derivation of side information from the inner decoder in a concatenated system. Pursley concludes by examining the use of side information in routing algorithms in packet radio networks.

5.5 Computer Memory

In Chapter 12, Saitoh and Imai describe the error control problems caused by faults in the integrated circuits used to control data flow in computers. The resulting error patterns in these systems are generally unidirectional; in other words, the erroneous bits have the same value. Saitoh and Imai show that this unidirectional tendency can be exploited to develop codes based on Reed-Solomon codes that outperform standard Reed-Solomon codes. Such advanced error control systems play an integral role in the development of extremely high-speed super computers.

6 DECODING REED-SOLOMON CODES

After the discovery of Reed-Solomon codes, a search began for an efficient decoding algorithm. None of the standard decoding techniques used at the time were very helpful. For example, some simple codes can be decoded through the use of a syndrome look-up table. The syndrome for a received word is computed, and the corresponding minimum-weight error pattern found in a table. This error pattern is subtracted from the received word, producing a code word. Unfortunately this approach is out of the question for all but the most trivial Reed-Solomon codes. For example, the (63,53) five-error-correcting Reed-Solomon code has approximately 10^{20} syndromes. The construction of a syndrome look-up table for this particular Reed-Solomon code is thus somewhat problematic.

In their 1960 paper, Reed and Solomon proposed a decoding algorithm based on the solution of sets of simultaneous equations (see Section 2). Though much more efficient than a look-up table, Reed and Solomon's algorithm is still useful only for the smallest Reed-Solomon codes. In the early 1960s, progress in the search for an efficient decoding algorithm was slow but steady. In 1960 Peterson provided the first explicit description of a decoding algorithm for binary BCH codes [13]. His "direct solution" algorithm is quite useful for correcting small numbers of errors but becomes computationally intractable as the number of errors increases. Peterson's algorithm was improved and extended to nonbinary codes by Gorenstein and Zierler (1961) [8], Chien (1964) [6], and Forney (1965) [7]. These efforts were productive, but Reed-Solomon codes capable of correcting more than six or seven errors still could not be used in an efficient manner. Detractors of coding research in the early 1960s had used the lack of an efficient decoding algorithm to claim that Reed-Solomon codes were nothing more than a mathematical curiosity. Fortunately the detractors' assessment proved to be completely wrong.

The breakthrough came in 1967 when Berlekamp demonstrated his efficient decoding algorithm for both nonbinary BCH and Reed-Solomon codes [1, 2]. Berlekamp's algorithm allows for the efficient decoding of dozens of errors at a time using very powerful Reed-Solomon codes. In 1968 Massey showed that the BCH decoding problem is equivalent to the problem of synthesizing the shortest linear feedback shift register capable of generating a given sequence [10]. Massey then demonstrated a fast shift register-based decoding algorithm for BCH and Reed-Solomon codes that is equivalent to Berlekamp's algorithm. This shift register-based approach is now commonly referred to as the *Berlekamp-Massey algorithm.*

In 1975 Sugiyama, Kasahara, Hirasawa, and Namekawa showed that Euclid's algorithm can also be used to efficiently decode BCH and Reed-Solomon codes [19]. Euclid's algorithm, named after its discoverer, the father of geometry, is a means for finding the greatest common divisor of a pair of integers. It can also be extended to more complex collections of objects, including certain sets of polynomials with coefficients from finite fields.

A detailed discussion of the various decoding algorithms for Reed-Solomon codes can be found in Chapter 5. Hasan, Bhargava, and Le-Ngoc begin this chapter by showing how various finite field operations can be implemented in software or hardware. They then proceed to the various decoding algorithms and describe the architectures through which these algorithms can be implemented. Hasan, Bhargava, and Le-Ngoc emphasize the techniques that allow for fast, efficient Reed-Solomon decoders, the very techniques that have made Reed-Solomon codes popular in so many different applications.

In Chapter 10 Berlekamp, Seroussi, and Tong describe a state-of-the-art Reed-Solomon decoder that is based on a hypersystolic architecture. This architecture is well-suited for very high-speed applications with sustained data rates that approach one gigabit per second. The hypersystolic architecture is described in detail, and it is shown how the algebraic algorithms in traditional RS decoding are adapted to it. The complexity of the cells in the hypersystolic array is limited to one Galois field multiplier, a few Galois field registers, and some simple control logic. Signal paths between cells are limited to two symbol-wide data lines and a few 1-bit control lines. Every signal, including control signals and the clock, has a fan-out and fan-in of only one line. The decoder supports codes of any length $n \leq 2^m - 1$, and redundancy $r < n/3$ over GF(2^m). A prototype of this decoder was built and demonstrated in 1987. This prototype decoder corrected all patterns of up to five errors per block in the (63,53) code mentioned earlier while running at a sustained data rate of 820 Mbits/s.

The "Holy Grail" of Reed-Solomon decoding research, to quote Cooper in Chapter 6, is the maximum-likelihood soft-decision decoder. A soft-decision decoder accepts analog values directly from the channel; the demodulator is not forced to decide which of the q possible symbols a given signal is supposed to represent. The decoder is thus able to make decisions based on the quality of a received signal. For example, the decoder is more willing to assume that a noisy value is indicating an incorrect symbol than that a clean, noise-free signal is doing so. All of the information on the "noisiness" of a particular received signal is lost when the demodulator assigns a symbol to the signal prior to decoding. It has been estimated that this results in a 2- to 3-dB loss in performance. Cooper examines the various means by which researchers are currently recovering some of this lost performance. He shows that the gap is closing but that there is still a great deal of work to be done before the Grail is found and the knights can go home.

7 THE FUTURE OF REED-SOLOMON CODES

The CD player is just the first of the many commercial, mass applications that Reed-Solomon codes can expect to enjoy in the coming years. The commercial world is becoming increasingly mobile, while simultaneously demanding reliable, rapid access to sales, marketing, and accounting information. Unfortunately the mobile channel is a nasty environment in which to communicate, with deep fades an ever-present phenomenon. Reed-Solomon codes are the single best solution; there is no other error control system that can match their reliability performance in the mobile environment. The optical channel provides another set of problems altogether. Shot noise and a dispersive, noisy medium plague line-of-sight optical systems, creating noise bursts that are best handled by Reed-Solomon codes. As optical fibers see increased use in high-speed multiprocessors, we can expect to see Reed-Solomon codes used there as well. In more specialized, single-use applications such as the occasional deep space probe, Reed-Solomon codes will continue to be used to force communication system performance ever closer to the line drawn by

Shannon. These are but a few of the applications showing that Irving Reed and Gus Solomon's codes have brought us a long way and that we can expect them to be with us for a very long time to come.

REFERENCES

[1] E. R. Berlekamp, "Nonbinary BCH Decoding," paper presented at the 1967 International Symposium on Information Theory, San Remo, Italy.

[2] E. R. Berlekamp, *Algebraic Coding Theory*, New York: McGraw-Hill, 1968. (Revised edition, Laguna Hills: Aegean Park Press, 1984.)

[3] R. E. Blahut, "Transform Techniques for Error Control Codes," *IBM Journal of Research and Development*, Volume 23, pp. 299–315, 1979.

[4] R. C. Bose and D. K. Ray-Chaudhuri, "On a Class of Error Correcting Binary Group Codes," *Information and Control*, Volume 3, pp. 68–79, March 1960.

[5] R. C. Bose and D. K. Ray-Chaudhuri, "Further Results on Error Correcting Binary Group Codes," *Information and Control*, Volume 3, pp. 279–290, September 1960.

[6] R. T. Chien, "Cyclic Decoding Procedure for the Bose-Chaudhuri-Hocquenghem Codes," *IEEE Transactions on Information Theory*, Volume IT-10, pp. 357–363, October 1964.

[7] G. D. Forney, "On Decoding BCH Codes," *IEEE Transactions on Information Theory*, Volume IT-11, pp. 549–557, October 1965.

[8] D. Gorenstein and N. Zierler, "A Class of Error Correcting Codes in p^m Symbols," *Journal of the Society of Industrial and Applied Mathematics*, Volume 9, pp. 207–214, June 1961.

[9] A. Hocquenghem, "Codes correcteurs d'erreurs," *Chiffres*, Volume 2, pp. 147–156, 1959.

[10] J. L. Massey, "Shift Register Synthesis and BCH Decoding," *IEEE Transactions on Information Theory*, Volume IT-15, Number 1, pp. 122–127, January 1969.

[11] J. L. Massey, "Deep Space Communications and Coding: A Match Made in Heaven," in *Advanced Methods for Satellite and Deep Space Communications*, J. Hagenauer (ed.), Lecture Notes in Control and Information Sciences, Volume 182, Berlin: Springer-Verlag, 1992.

[12] R. J. McEliece, *Finite Fields for Computer Scientists and Engineers*, Boston: Kluwer Academic, 1987.

[13] W. W. Peterson, "Encoding and Error-Correction Procedures for the Bose-Chaudhuri Codes," *IRE Transactions on Information Theory*, Volume IT-6, pp. 459–470, September 1960.

[14] E. Prange, "Cyclic Error-Correcting Codes in Two Symbols," Air Force Cambridge Research Center-TN-57-103, Cambridge, Mass., September 1957.

[15] E. Prange, "Some Cyclic Error-Correcting Codes with Simple Decoding Algorithms," Air Force Cambridge Research Center-TN-58-156, Cambridge, Mass., April 1958.

[16] E. Prange, "The Use of Coset Equivalence in the Analysis and Decoding of Group Codes," Air Force Cambridge Research Center-TR-59-164, Cambridge, Mass., 1959.

[17] I. S. Reed and G. Solomon, "Polynomial Codes over Certain Finite Fields," *SIAM Journal of Applied Mathematics*, Volume 8, pp. 300–304, 1960.

[18] R. C. Singleton, "Maximum Distance Q-nary Codes," *IEEE Transactions on Information Theory*, Volume IT-10, pp. 116–118, 1964.

[19] Y. Sugiyama, Y. Kasahara, S. Hirasawa, and T. Namekawa, "A Method for Solving Key Equation for Goppa Codes," *Information and Control*, Volume 27, pp. 87–99, 1975.

[20] M. A. Tsfasman, S. G. Vladut, and T. Zink, "Modular Curves, Shimura Curves and Goppa Codes Which Are Better Than the Varshamov-Gilbert Bound," *Mathematische Nachrichten*, Number 109, pp. 21–28, 1982.

[21] S. B. Wicker, *Error Control Systems for Digital Communication and Storage*, Englewood Cliffs, N.J.: Prentice-Hall, 1994.